DUAL IMPORTANCE WEIGHT GAN

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative Adversarial Networks (GAN) are trained to generate a sample image of interest. To this end, generative network of GAN learns implicit distribution of true dataset from the classification samples with candidate generated samples. However, in real implementation of GAN, training the generative network with limited number of candidate samples guarantees to properly represent neither true distribution nor the distribution of generator outputs. In this paper, we propose dual importance weights for the candidate samples represented in the latent space of auto-encoder. The auto-encoder is pre-trained with real target dataset. Therefore, the latent space representation allows us to compare real distribution and the distribution of generated samples explicitly. Dual importance weights iteratively maximize the representation of generated samples for both distributions: current generator outputs and real dataset. Proposed generative model not only resolves mode collapse problem of GAN but also improves the convergence on target distribution. Experimental evaluation shows that the proposed network learns complete modes of target distribution more stable and faster than state of the art methods.

1 INTRODUCTION

Recently, generative networks have been widely studied thanks to the explosive and successful applications of Generative Adversarial Networks (GAN) proposed by Goodfellow et al. (2014). Main difficulty of training step in the generative model is evaluating high-dimensional original probability distribution. Estimation of the probability distribution of target dataset from sparse samples is not a trivial task that plays a critical role in the quality of generated samples.

GAN is composed of two main networks: discriminator and generator. Core idea is learning the target distribution through two-player minimax game theory. The discriminator helps generator to learn the representation of the target distribution by distinguishing the difference between generated and real samples. As a result, generator is able to produce samples which resemble target real data. Although GAN has been successfully implemented in many applications, it suffers from ill-training problems such as oscillation between modes, mode collapsing, etc. Especially, mode collapsing results in the generation of samples only from a single or a few modes of target data distribution losing diversity of target dataset. Main reason of mode collapsing problem is that the discriminator is incapable of delivering any information about samples' diversity. Once the generator finds optimal point of a fixed discriminator in each generator training step, the network produces same samples driving the expectation value in objective function becomes minimum regardless of input noise vectors. Consequently, generator repeatedly generates certain good instance rather than diverse instances from entire data distribution.

Metz et al. (2016) proposes unrolled GAN that trains generator using unrolled multiple samples rather than single target at the generator output, which alleviate mode collapsing. VEEGAN in Srivastava et al. (2017) employs a reconstruction network which is not only mapping real data distribution to a Gaussian in encoded space but also approximating reverse action of the generator. Because VEEGAN estimates implicit probability of real dataset, it prevents mode collapsing problem and produces more realistic samples. Bang & Shim (2018) propose a manifold guided generative adversarial network(MGGAN) for guiding a generator by adding another adversarial loss in manifold space transformed by pre-trained encoder network. This enables the generator to learn all modes of target distribution without impairing the quality of image. Although real data distribution is represented in the manifold loss of their second discriminator, it is not able to estimate explicit distance between the distributions. OT-GAN(Salimans et al. (2018)) expands generator loss function using

optimal transport theory. They define a new metric, $MinibatchEnergyDistance$, over probability distribution in a adversarially learned feature space. Although they make the condition of GAN to be more stable than other state-of-the-art research, the computational cost of their large mini-batches is expensive to be practical in real applications.

In all such previous approaches, using increased number of samples or larger mini-batches for the training of generator network allows more stable and improved performance. On the other hand, they require hugely increased computation cost and training time. Our idea is that given limited number of generated samples in the training step, representation ability of the samples can be evaluated and maximized for both distributions (current generator outputs and real dataset) adopting importance weights estimation of the samples.

In this paper, we propose dual importance weights for the generated candidate samples in generator network training step represented in the latent space of auto-encoder. The auto-encoder is pre-trained with real target dataset. We assume that auto-encoder trained with target dataset has ability to represent the distribution of target dataset in the latent space under optimally minimized dimensions. Therefore, the latent space representation allows us to compare real distribution and the distribution of generated samples explicitly. Our dual importance weights on generated samples iteratively maximize the representation ability for the generator and guide the generator to find complete modes of real data distribution. To this end, we expand generator objective function to save the diversity of target distribution via adopted auto-encoder which works as a bridge between data and manifold space. As a result, proposed dual importance weight GAN not only resolves mode collapse problem of GANs but also improves the convergence on target distribution. Transformed distributions to the latent space enables explicit representation of generated and real data. We calculate actual distance between the two distributions and achieve fast and robust convergence to optimal states avoiding mode collapse problem.

2 GAN WITH AUTO-ENCODER

GAN is motivated by game theory that two players (generator and discriminator) compete with each other in a zero-sum game framework. The generator learns the target distribution via an adversarial training process with discriminator. When the generator transform a noise vector z into a data vector $G(z)$ on target space, the discriminator tries to estimate the probability if input sample comes from target distribution $P(x)$ or not. At the end of the training process, the generator produces outputs of $P(x)$. GAN defines this adversarial problem as follow:

$$\min_G \max_D L_{GAN}(D, G) = \mathbb{E}_{x \sim P_{data}} \log D(x) + \mathbb{E}_{z \sim P_z} \log(1 - D(G(z))) \tag{1}$$

where the goal of GAN is to find the optimal parameter set of G and D in equation (1) minimizing the generator loss and maximizing the discriminator loss.

Auto-encoder is a neural network that learns optimal representation of unlabeled input data in encoded space. It generates original input through decoding network. Auto-encoder not only decreases the dimensionality of input data but also finds optimal abstraction of input data for better discrimination between subjects. Generative networks with auto-encoder and adversarial learning algorithms have been proposed to improve GAN. Adversarial auto-encoder (AAE) Makhzani et al. (2015) is a type of auto-encoder combined with a discriminator and learns target data distribution using both reconstruction and adversarial losses. The adversarial loss let the encoder learn a posterior which follows imposed prior during adversarial training. More recently, Rosca et al. (2017) suggests a α-GAN which combines variational auto-encoder (VAE, Kingma & Welling (2013)) and GAN. Using reconstruction and adversarial losses, α-GAN tries to address mode collapse problem of GAN producing blurry images of VAE. Proposed generative network extends original GAN Goodfellow et al. (2014) with auto-encoder. Auto-encoder trained with real dataset optimally reduces the dimension of the dataset. In the encoded space, auto-encoder extracts essential feature set to represent all modes and aspects of original data distribution which is ideal to guide generative network.

3 SAMPLING WITH DUAL IMPORTANCE WEIGHT

Based on our GAN with auto-encoder, obtaining good samples is a critical for improved performance. Importance sampling is a statistical technique for estimating characteristics of distribution

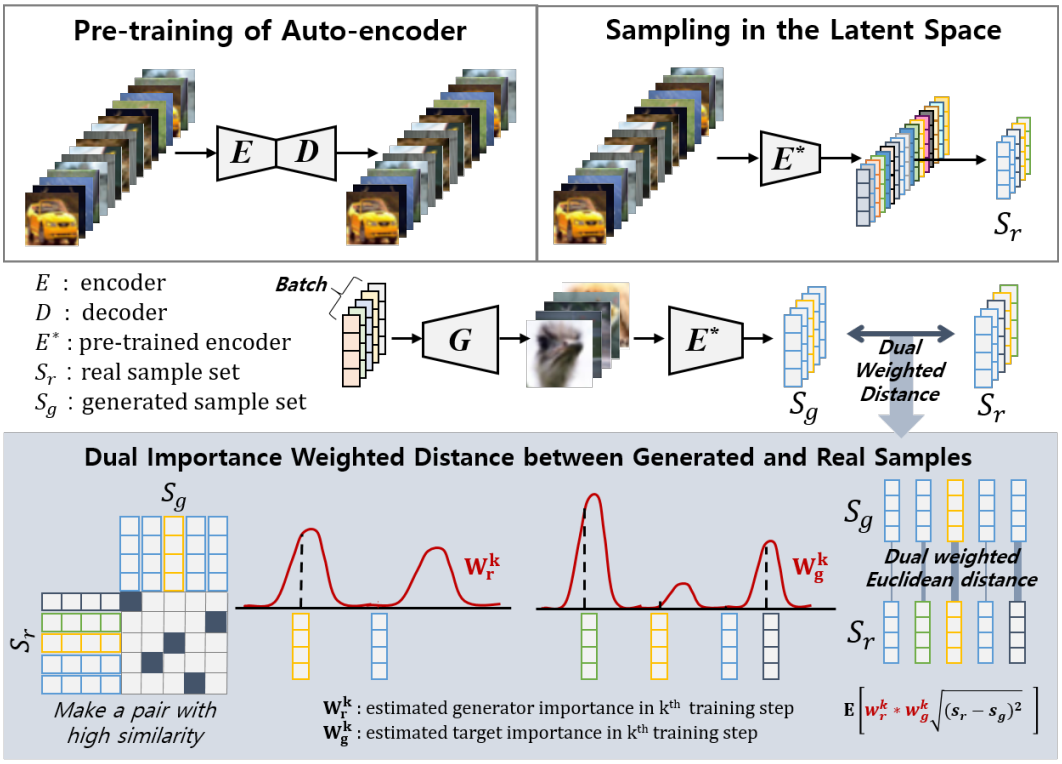


Figure 1: Proposed generative network with dual importance weights

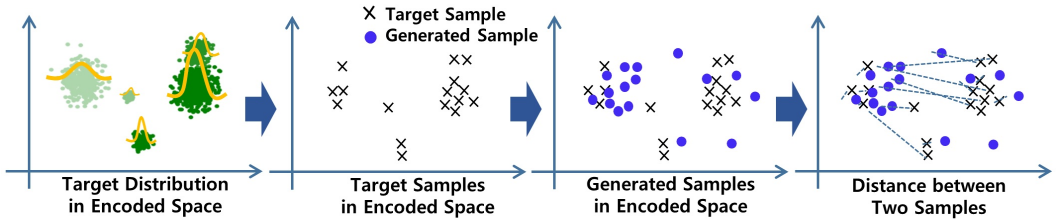


Figure 2: Distance calculation between generated and real data samples in encoded space

with the samples generated from different distribution. The basic idea of importance sampling is to give weights to the samples for the proper representation of target distribution. This weight adjusts the contribution of each sample to the representation so that we are able to improve the approximation quality. In GAN, we assume that current generated samples represent the distribution of complete outputs of the generator. And we expect the generated samples ultimately represent the distribution of real data. Diesendruck et al. (2018) proposes importance weighted generative networks with modified adversarial loss function from differently contributing samples of mini-batch. Importance weighted auto-encoders (IWAE; Burda et al. (2015)) is a variant of variational auto-encoder (VAE). An improved loss function of log-likelihood lower bound derived from importance weighting has tighter boundary than VAE.

We define a new distance $D_a(s_r, s_g)$ which measures the distance between paired real and generated samples in the latent space of auto-encoder.

$$D_a(s_r, s_g) = w_r^k w_g^k \sqrt{(s_r - s_g)^2} \tag{2}$$

where $s_r \in S_r$, $s_g \in S_g$ denote samples from real and generated sample sets in encoded space. Real sample s_r is extracted from the distribution of real data in the encoded space and the the distribution of real data is obtained using Kernel density estimation (KDE) on the complete real dataset transformed to the encoded space of pre-trained auto-encoder.

We define the expectation of euclidean distance over mini-batch samples as the distance between real and generated samples. w_r^k and w_g^k are target importance weight and generator importance weight, respectively. Target importance weight is assigned to each target real sample and generator importance weight is assigned to each generated sample. First, generator importance weight w_g indicates how well current generated sample represents generator outputs. In order to obtain the weight for each sample, we accumulate past N generated sample sets constructing the distribution of generator outputs using Kernel density estimation (KDE). A generated sample of high generator importance weight contributes more to the distance calculation in equation (2). When the current generated samples poorly represent the current generator distribution, they gets smaller weights and are ignored in the training. In this manner, iterative sample quality evaluation makes stable environment of GAN training procedure. Secondly, target real importance weight w_r indicates how well all generated samples contribute to the estimation of current real data sample. We assign the weight for each target real sample based on the degree of how much the real sample is covered by generated samples. Calculated distance of a real sample from paired generated samples in the past iterations add importance to the real sample. In other words, if a real sample has higher distances to all generated samples during the past training steps, the real sample gets higher real importance weight. And then, in the next training step, network is updated to generated this isolated real sample due to the higher loss value. Finally, both importance weights encourage generated samples to cover entire target real data samples. At each training step, both weights are updated adjusting the importance of generated and target samples. Due to the elaborate evaluation of the generated sample quality at each training step, proposed network converges to target real distribution faster without mode collapse problem.

Our generator loss function is shown in equation (3). If GAN unexpectedly generates samples from single mode of real dataset, second term in equation (3) increases forcing the network to learn other modes.

$$\min_G -\mathbb{E}_{z\sim P_z} \log D(G(z)) + \alpha \times \mathbb{E}_{s_r\sim S_r, s_g\sim S_g} D_a(s_r, s_g) \tag{3}$$

Overall training procedures are summarized in Figure 6. First we train auto-encoder with target real dataset. The target data distribution in the encoded space is estimated by KDE(Kernel density estimation) from entire dataset transformed to the encoded space. And then, we obtain same number of good real samples of batch size which reflects target distribution. Note that the real samples are extracted once and used for all iterations of the training. For optimal generated and real sample pair matching in distance D_a calculation, we calculate distances among all real samples to all generated samples and assign pair one by one minimizing average distance. Based on generated candidate samples, our approach tries to find all aspects of real distribution qualifying generated samples. Better generated samples evaluated by generator importance weight understand generator better. Better target importance understands target real dataset better. Finally, our method let the generator understand target real dataset better.

4 Experimental Evaluation

We perform quantitative and qualitative evaluation on three datasets: Mixture of Gaussians, (stacked) MNIST, and Cifar10. Several similarity metrics are used to quantify generation performance.

4.1 Mixture of Gaussians

We have created several test distributions made of Gaussians: mixture of eight 2D Gaussians located in a ring, 25 Gaussians located in a grid and 25 Gaussians randomly located, 27 Gaussians in a 3D cube. 25 random Gaussians are anisotropic. VEEGAN(Srivastava et al. (2017)) and Unrolled GAN(Metz et al. (2016)) are compared with our method. Identical network architecture is used for fair comparison. Generator has three layers of fully connected MLPs with 128 nodes without dropout and batch normalization and discriminator has two layers of fully connected MLPs without dropout and batch normalization. We choose the dimension of encoded space of auto-encoder that properly reproduces original data. Following Srivastava et al. (2017), we employ two metrics for quantitative evaluation. First, number of modes found are counted. Secondly, the quality of samples are measured by high quality sample ratio (HQS). However it's hard to evaluate how the generated samples are able to cover the entire real distribution just using number of modes and percentage

of high quality samples. Thus we measure the distance between estimated target and generated distribution. We map the sample to canonical space and count the number of points within each canonical unit. By measuring Jensen-Shannon divergence(JSD) between the distributions (P_r, P_g), we evaluate how well generator follows target distribution. Figure 3 shows results on our mixture of Gaussians dataset. Proposed method outperforms over two state of the art methods showing much faster convergence without mode collapse problem. Table 1 summarizes quantitative results showing outstanding performance of our generative network.

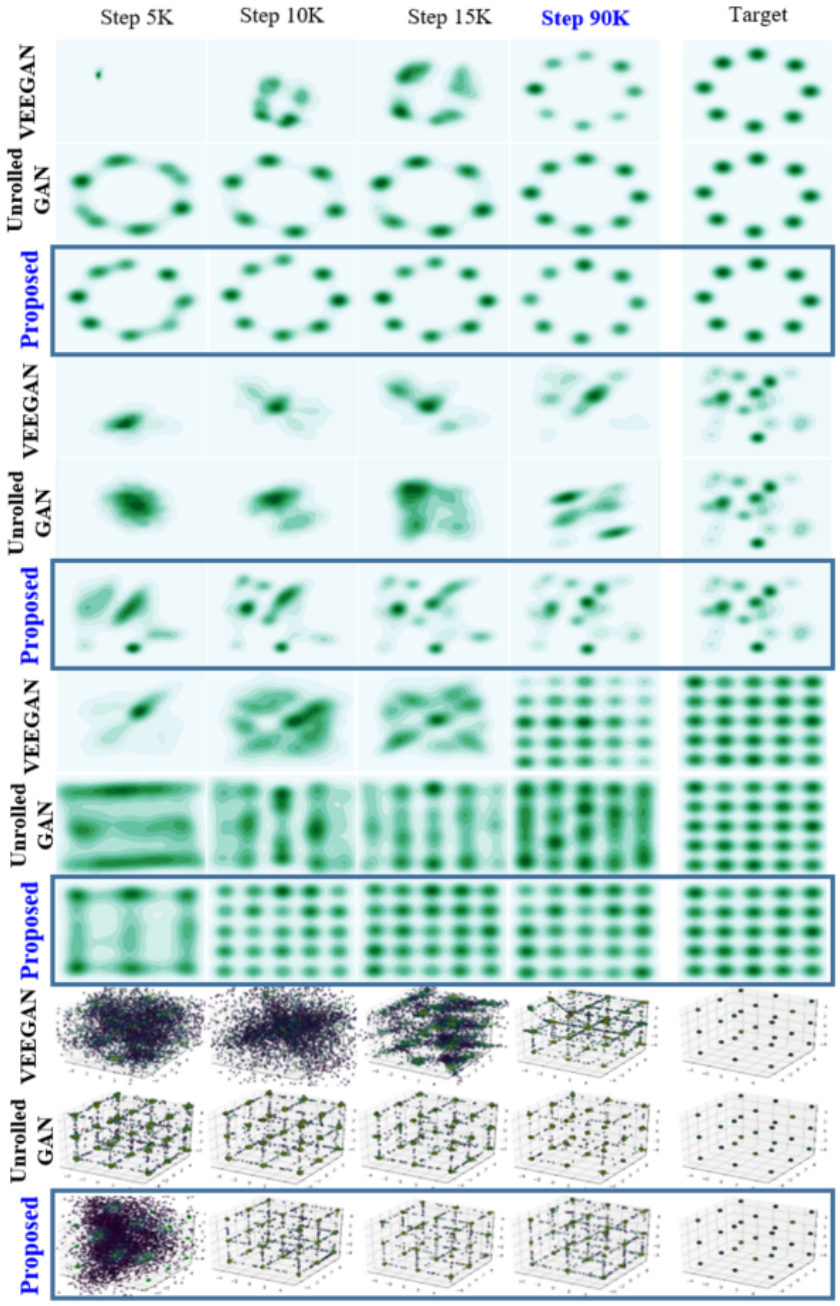


Figure 3: Experimental results compared to VEEGAN(Srivastava et al. (2017)) and Unrolled GAN(Metz et al. (2016)): 2D Ring, 2D Grid(isotropic and unisotropic), and 3D Cube testset. Proposed method converges faster than state-of-the-art methods.

Table 1: Quantitative Evaluation: Number of modes found, HQS(High Quality Sample), and JSD(Jensen-Shannon divergence) between real and generated sample distributions. Best result is indicated bold face font.

	METRIC	Unrolled GAN(std)	VEEGAN(std)	**Proposed**(std)
2D Ring	Modes(Max 8)	8(0)	8(0)	**8**(0)
	% HQS	30.9(0.006)	60.4(0.005)	**85.5**(0.006)
	JSD(real‖generated)	0.254(0.004)	0.19(0.005)	**0.172**(0.004)
2D Grid (uniform)	Modes(Max 25)	25(0)	24.1(0.31)	**25**(0)
	% HQS	14.4(0.006)	65.4(0.006)	**82.5**(0.004)
	JSD(real‖generated)	0.47(0.006)	0.21(0.004)	**0.12**(0.004)
2D Grid (random)	Modes(Max 25)	24.4(0.5)	24.3(0.637)	**25**(0)
	% HQS	15.9(0.005)	63.7(0.007)	**83.4**(0.006)
	JSD(real‖generated)	0.38(0.007)	0.32(0.007)	**0.15**(0.004)
3D Cube	Modes(Max 27)	27(0)	26.6(0.5)	**27**(0)
	% HQS	**85.0**(0.348)	43.3(0.007)	80.0(0.005)
	JSD(real‖generated)	0.194(0.004)	0.31(0.005)	**0.125**(0.004)

4.2 MNIST

In this experiment, we expand MoG dataset to image space with MNIST and stacked MNIST. Evaluation on MNIST data (Figure 4) shows that proposed method generates more number of digits than original GAN. Stacked MNIST is designed for high complexity evaluation with extended number of modes. Stacked MNIST is synthesized by stacking different MNIST digits in respective colors. This synthesized dataset has 1000 modes which are the combinations of 10 classes in 3 channel. We use implemented generator architecture of standard DGGAN. For the discriminator, we use same architecture with DCGAN and Unrolled GAN. For VEEGAN, the architecture of discriminator is designed following Srivastava et al. (2017). Auto-encoder has three convolutional layers with 5 by 5 filters, 2 fully connected layers for encoder part and one fully connected layer, 3 transposed convolutional layers are used for decoder. First, we train the classifier using MNIST dataset which assigns the mode to generated images. In this evaluation, we use 20,000 generated samples and they are given the mode from pre-trained classifier. For example, one sample has 3 channels which represent different digit. Digit for each sample in channel is determined by MNIST classifier.

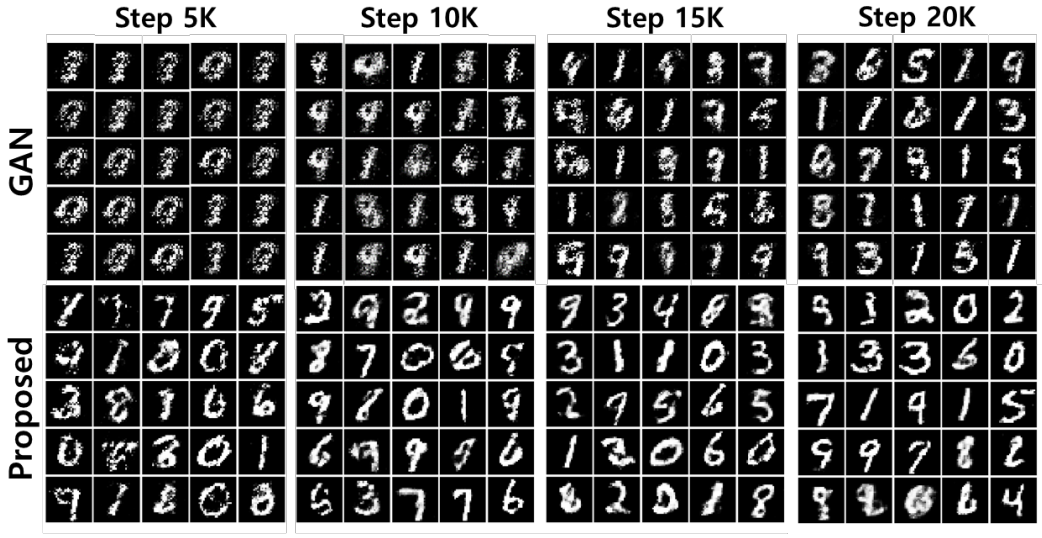
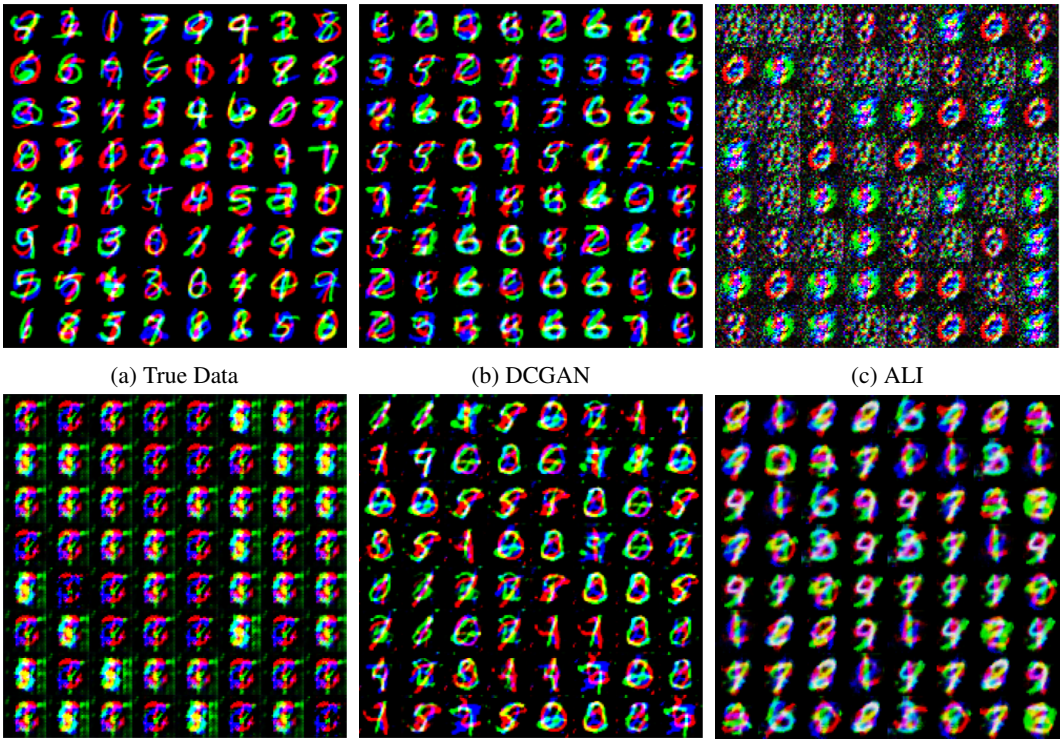


Figure 4: Experimental results on MNIST dataset. First row is original GAN and second row is proposed method.



(a) True Data (b) DCGAN (c) ALI

(d) Unrolled (e) VEEGAN (f) Proposed

Figure 5: Generated stacked MNIST samples from trained models. We refer these images from Srivastava et al. (2017) paper except our result.

(a) VEEGAN (b) DCGAN (c) Proposed

Figure 6: Experimental results are compared to VEEGAN and DCGAN using CIFAR 10 dataset: VEEGAN and DCGAN frequently suffer from mode collapsing in yellow circles.

4.3 CIFAR 10

We also evaluate our method using CIFAR-10 dataset which includes 32x32 color images with 10 classes collected by Krizhevsky & Hinton (2009). The architecture is same to 4.2. Generated results of VEEGAN and DCGAN are collected from Srivastava et al. (2017). As shown in Figure 6, VEEGAN and DCGAN frequently include identical samples suffering from mode collapsing (see yellow circles).

5 CONCLUSION

In this paper, we propose dual importance weights for the candidate samples represented in the latent space of auto-encoder. Dual importance weights iteratively maximize the representation of generated samples for both distributions: current generator outputs and real dataset. Evaluation and comparison are extensively performed on three datasets showing promising performance of the proposed method. On the other hand, our method involves additional computation for sample pair matching. Structure of auto-encoder and the dimension of encoded space can be further optimized for improved quality of generated samples.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

Duhyeon Bang and Hyunjung Shim. Mggan: Solving mode collapse using manifold guided training. *arXiv preprint arXiv:1804.04391*, 2018.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Maurice Diesendruck, Ethan R Elenberg, Rajat Sen, Guy W Cole, Sanjay Shakkottai, and Sinead A Williamson. Importance weighted generative networks. *arXiv preprint arXiv:1806.02512*, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, and Shakir Mohamed. Variational approaches for auto-encoding generative adversarial networks. *arXiv preprint arXiv:1706.04987*, 2017.

Tim Salimans, Han Zhang, Alec Radford, and Dimitris Metaxas. Improving gans using optimal transport. *arXiv preprint arXiv:1803.05573*, 2018.

Akash Srivastava, Lazar Valkoz, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems*, pp. 3308–3318, 2017.